RAJALAKSHMI
ENGINEERING COLLEGE
An AUTONOMOUS Institution
Affiliated to ANNA UNIVERSITY, Chennai

TRAIN DELAY PREDICTION USING LOGISTIC REGRESSION

Submitted by

Gopica H(231801043)

Hannah James(231801047)

AI23331 - FUNDAMENTALS OF MACHINE LEARNING

Department of Artificial Intelligence and Data Science

Rajalakshmi Engineering College, Thandalam Nov 2024

RAJALAKSHMI
ENGINEERING COLLEGE

BONAFIDE CERTIFICATE

NAME……………………………………………………………………………………….

ACADEMIC YEAR…………………SEMESTER…………. BRANCH …………….

UNIVERSITY REGISTER No.

Certified that this is the Bonafide record of work done by the above students in the Mini Project titled "**TRAIN DELAY PREDICTION USING LOGISTIC REGRESSION**" in the subject **AI23331– FUNDAMENTALS OF MACHINELEARNING** during the year **2024 - 2025.**

Signature of Faculty – in – Charge

Submitted for the Practical Examination held on -------------------------------

Internal Examiner **External Examiner**

TABLE OF CONTENTS

ABSTRACT

Train delays cause significant disruptions to passenger schedules and complicate transportation logistics, making it crucial for railway operators to predict delays and optimize scheduling. This project addresses this challenge y developing a machine learning model using logistic regression to predict train delays. The model considers several key factors—distance between stations, weather conditions, day of the week, time of day, train type, and route congestion—that can influence train punctuality. Each of these factors is processed to improve prediction accuracy, such as encoding weather conditions and route congestion levels to reflect their varying impacts on delays.

The data undergoes preprocessing steps to prepare it for modelling. Categorical variables, like the day of the week and train type, are one-hot encoded, while numeric features, such as distance and congestion levels, are standardized. This ensures that no single feature disproportionately affects the model. The logistic regression model is then trained to identify patterns between the input features and the likelihood of a train being delayed. The model outputs a probability score, which is used to classify a train as either "On Time" or "Delayed" based on a predefined threshold.

To evaluate model performance, key metrics such as accuracy, precision, recall, and a confusion matrix are used. These metrics provide insights into the model's effectiveness at predicting delays and highlight areas for improvement. The results show that the logistic regression approach can successfully identify delay patterns, helping railway operators anticipate potential delays and take preventive actions to optimize scheduling. This predictive model offers a reliable, cost-effective, and interpretable solution for enhancing train scheduling and improving passenger satisfaction.

CHAPTER 1
INTRODUCTION

1.1 GENERAL

Train delay prediction is crucial for optimizing railway operations and improving service efficiency and customer satisfaction. Delays in train schedules can cause significant disruptions for passengers and freight services, resulting in economic losses and operational inefficiencies. Thus, developing a predictive model for train delays is essential for railway networks aiming to improve punctuality and reliability. This project focuses on creating a delay prediction system using logistic regression, a well-established classification algorithm, known for its simplicity and interpretability.

1.2 NEED FOR THE STUDY

Railway operators face challenges in ensuring the punctuality of train services, with delays often caused by various factors such as weather conditions, congestion, and route issues. These delays can lead to significant disruptions for passengers and freight services, affecting operational efficiency and leading to economic losses. By predicting train delays, operators can take proactive measures to mitigate disruptions, optimize resource allocation, and improve service reliability. This study aims to develop a predictive model that can accurately forecast train delays, contributing to enhanced decision-making, improved operational efficiency, and customer satisfaction in railway networks.

1.3 OBJECTIVE OF THE PROJECT

The primary objective of this project is to develop a predictive model that can accurately forecast train delays based on various influencing factors. The model will utilize historical data, including features such as distance between stations, weather conditions, time of day, day of the week, train type, historical delay records, and route congestion. By applying logistic regression, this project aims to provide a reliable and interpretable tool for railway operators to predict delays and optimize train scheduling, ultimately enhancing service reliability and operational efficiency.

1.4 OBJECTIVE OF THE STUDY

This study has several specific objectives:

- To analyze historical train journey data, identifying key features such as weather conditions, train type, and route congestion that influence train delays.
- To develop a predictive model using logistic regression, optimizing its parameters to maximize accuracy and minimize overfitting.
- To evaluate model performance using metrics such as accuracy, precision, recall, and F1-score, along with a confusion matrix.
- To provide insights into the relative importance of each feature influencing train delays, offering valuable interpretability for railway operators.
- To compare logistic regression with alternative predictive algorithms, such as decision trees and random forests, to validate its effectiveness and identify potential areas for improvement.
- To document the entire process, ensuring transparency and reproducibility, and contributing valuable insights for future research in predictive analytics for train scheduling.

ALGORITHM USED

The logistic regression algorithm used in this project is a widely recognized and effective method for binary classification tasks, particularly well-suited for predicting train delays. Logistic regression works by modelling the probability of a binary outcome, in this case, whether a train will be “On Time” or “Delayed,” based on a set of input features. These features include factors such as distance between stations, weather conditions, day of the week, time of day, train type, and route congestion, all of which can influence train punctuality. The model outputs a probability score, which is then thresholded to classify the train's status as either "On Time" or "Delayed."

In the context of train delay prediction, data preprocessing is a critical step. Categorical variables, such as the day of the week and train type, are one-hot encoded to allow the model to process them effectively. Numeric features, such as distance and congestion levels, are standardized to ensure that no single feature disproportionately influences the model. The logistic regression model is then trained on the processed data to learn the relationship between these factors and the likelihood of a delay. Various evaluation metrics, including accuracy, precision, recall, and confusion matrix, are used to assess model performance and ensure its effectiveness in real-world applications.

The model’s performance is validated using cross-validation techniques and compared with other models to ensure robustness and reliability. The logistic regression approach proves effective at identifying delay patterns and offering actionable predictions for railway operators. By providing a cost-effective and interpretable tool, this model helps optimize train scheduling, reduce delays, and improve passenger satisfaction. Future extensions of the project could incorporate additional data features, such as real-time information or historical maintenance records, to further improve the model’s accuracy and robustness, enhancing its predictive capabilities for real-time ma

CHAPTER 2
SYSTEM ARCHITECTURE

HARDWARE REQUIREMENTS

Development and Training

- Processor: Dual-core (Intel i5 or AMD equivalent) or higher; quad-core preferred.
- RAM: 8 GB recommended; 4 GB minimum.
- Storage: 256 GB SSD or HDD; SSD preferred for speed.
- GPU: Not required (optional for experimenting with other models).

Testing and Evaluation

- Processor: Dual-core or quad-core.
- RAM: 4-8 GB.
- Storage: 100 GB HDD or SSD.

Deployment

- Cloud Server: AWS, Google Cloud, or Azure recommended.
- Local Server:
  - Processor: Quad-core or higher.
  - RAM: 8 GB or higher.
  - Storage: 100 GB.
- Edge Device (Optional): Raspberry Pi for on-site predictions with optimized models**.**

SOFTWARE REQUIREMENTS

Software Requirements (Summary)

- OS: Windows 10/11, macOS, or Linux (e.g., Ubuntu)
- Language: Python 3.x
- IDE: Jupyter Notebook, PyCharm, or VS Code

Libraries

- Data: Pandas, NumPy
- Visualization: Matplotlib, Seaborn
- Machine Learning: scikit-learn

CHAPTER 3
SYSTEM OVERVIEW

3.1SYSTEM ARCHITECTURE

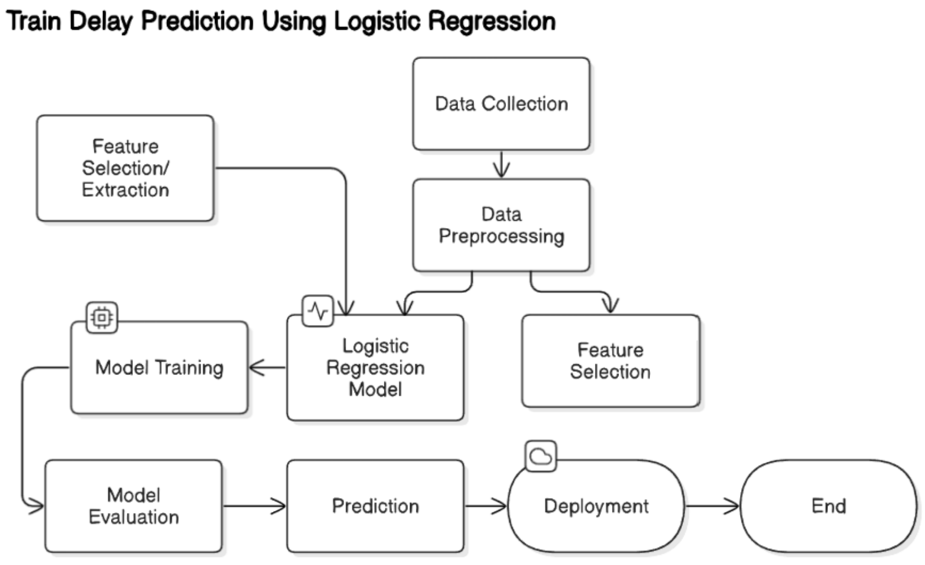


FIGURE 3.1.1: SYSTEM ARCHITECTURE DIAGRAM

The architecture of the train delay prediction model is primarily built around logistic regression, a robust and interpretable classification algorithm. This model aims to predict the likelihood of a train being delayed based on several key input features, such as weather conditions, route congestion, day of the week, time of day, train type, and distance between stations. Though simpler than more complex models like deep neural networks, logistic regression offers an efficient and interpretable approach for predicting binary outcomes (on-time or delayed) while maintaining reasonable accuracy.

Data Preprocessing

Before training the logistic regression model, the data undergoes rigorous preprocessing to ensure optimal input quality. Categorical features, including the day of the week, time of day, and train type, are converted into numerical representations using one-hot encoding. This ensures that the model can process these variables without assuming any ordinal relationship. Numeric features such as distance and route congestion are standardized to ensure that no single feature disproportionately affects the model due to differing scales. Feature scaling is achieved using techniques like StandardScaler, which transforms the features to have zero mean and unit variance. This is crucial because logistic regression is sensitive to feature scaling, and standardization improves the model's convergence during training.

Feature Engineering

To enhance the model's predictive power, certain features are engineered to reflect more meaningful representations of the problem. For example, weather conditions and route congestion are encoded numerically to capture their impact on delays. These features are designed to reflect their respective effects on train punctuality: weather (clear, rainy, or foggy) is encoded with increasing severity, and route congestion (low, medium, or high) is represented similarly.

Additionally, historical delays are used as a proxy for potential future delays. A binary target variable is created, where delays greater than a threshold (e.g., 10 minutes) are classified as "Delayed," and delays below this threshold are considered "On Time." This binary classification makes it easier for the logistic regression model to output a probability score that can be converted into an actionable decision (whether the train is likely to be on time or delayed).

Logistic Regression Model

At the core of the model is logistic regression, a statistical method used for binary classification. Logistic regression estimates the probability of an event (train delay) occurring based on the input features. The model uses the logit function to map the weighted sum of input features to a probability value between 0 and 1. These weights are learned during training using an optimization technique (typically Gradient Descent) that minimizes the log-loss (cross-entropy) function.

The logistic regression model outputs a probability score representing the likelihood of a train being delayed. This score is then converted into a binary classification, where a predefined threshold (e.g., 0.5) determines whether the model classifies the train as "On Time" or "Delayed."

Model Training and Evaluation

The model is trained on historical data using the training dataset, and its performance is assessed using accuracy, precision, recall, and F1-score metrics. Additionally, a confusion matrix is employed to visually assess how well the model differentiates between delayed and on-time trains. The model's hyperparameters, such as the regularization strength, can be tuned to ensure that it avoids overfitting while maintaining generalizability.

To ensure a fair evaluation, the dataset is split into a training set and a test set (e.g., 80% for training and 20% for testing). The model's ability to predict unseen data is tested on the test set, and performance metrics are derived from the predicted versus true labels.

Model Effectiveness

The logistic regression model is efficient and interpretable, making it particularly useful for applications where understanding the decision-making process is crucial. While the model may not capture highly complex non-linear relationships like some machine learning models (e.g., neural networks), it provides a transparent and reliable tool for predicting train delays. The simplicity of logistic regression allows railway operators to quickly integrate the model into existing systems and take real-time actions to address potential delays.

Conclusion

In summary, the architecture of the train delay prediction system, built around logistic regression, leverages effective data preprocessing, feature engineering, and robust evaluation techniques. By utilizing a binary classification approach, the model outputs probability scores to predict train delays based on multiple influential features. Although relatively simple compared to more complex algorithms, logistic regression remains highly effective and interpretable, offering a valuable tool for improving scheduling accuracy and operational efficiency in train management systems. The model's straightforward design also facilitates integration into real-time systems, supporting timely interventions to optimize train schedules.

3.1 MODULE 1 – DATA COLLECTION AND PREPROCESSING

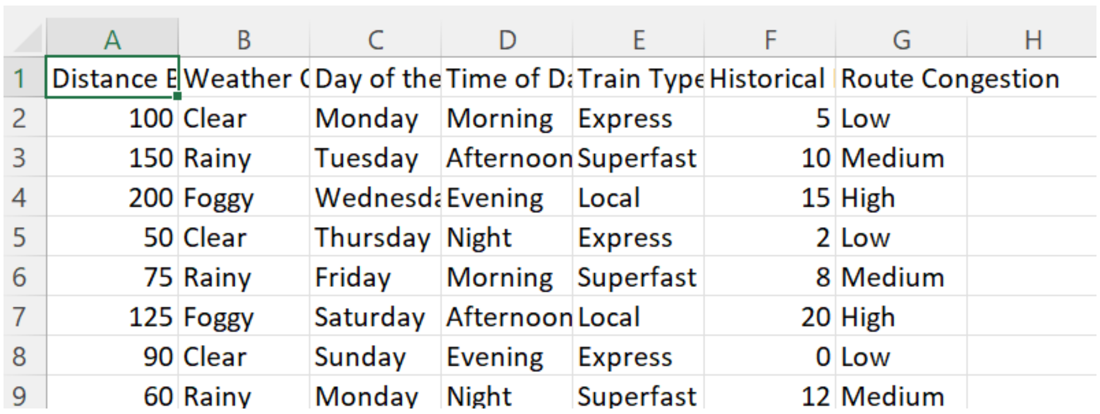
Data Preparation:

Download Dataset:

The first step is to download the train delay dataset, which is available on platforms like Kaggle. This dataset typically contains historical train information, including timestamps, distance between stations, weather data, route congestion levels, and whether the train was delayed or on time.

Data set

The dataset used in this project contains information about train schedules and factors that might cause delays. It includes the Train ID, Departure and Arrival Times, and the Distance between stations. It also has details on Weather Conditions, such as clear skies or rain, which can affect train performance. Congestion Levels show how crowded the train is, and the Route or line the train takes is recorded. The Day of the Week is also included, as some days may have more delays than others. The Delay column is the target variable, indicating if a train was delayed (1) or on time (0). This dataset helps in predicting train delays based on these factors.

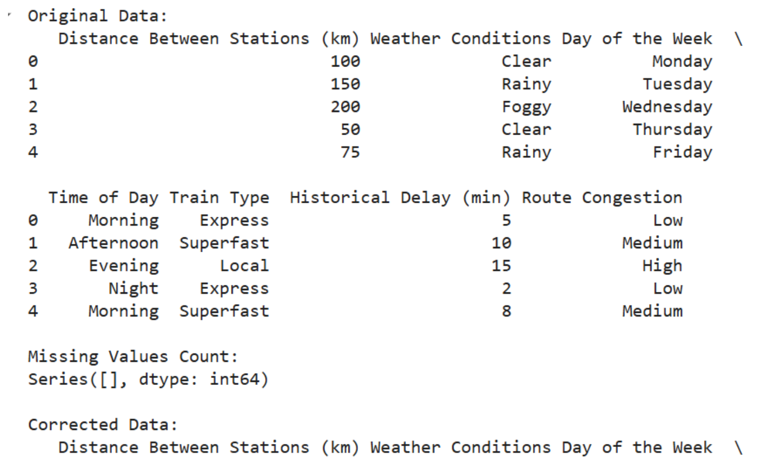


	A	B	C	D	E	F	G	H
1	Distance E	Weather (	Day of the	Time of D:	Train Type	Historical	Route Congestion	
2	100	Clear	Monday	Morning	Express	5	Low	
3	150	Rainy	Tuesday	Afternoon	Superfast	10	Medium	
4	200	Foggy	Wednesda	Evening	Local	15	High	
5	50	Clear	Thursday	Night	Express	2	Low	
6	75	Rainy	Friday	Morning	Superfast	8	Medium	
7	125	Foggy	Saturday	Afternoon	Local	20	High	
8	90	Clear	Sunday	Evening	Express	0	Low	
9	60	Rainy	Monday	Night	Superfast	12	Medium	

FIGURE 3.1.2: INPUT THROUGH CSV FILE

1.Preprocessing:

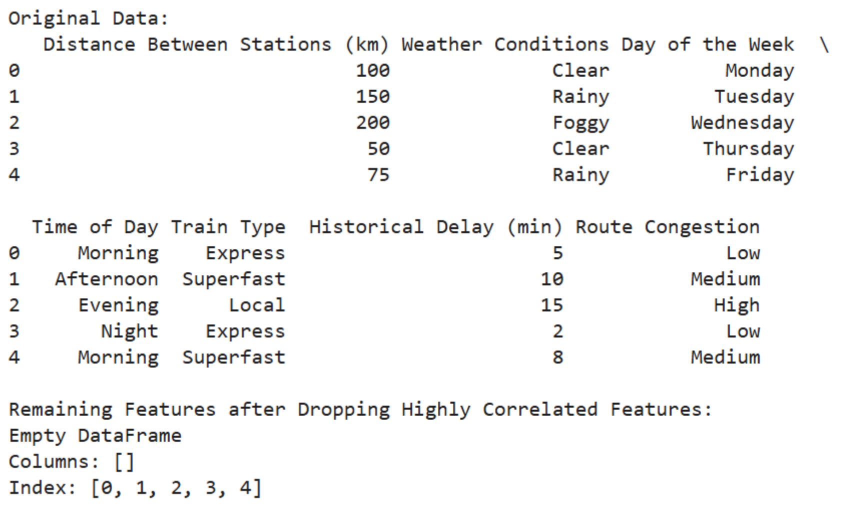
Next, we preprocess the dataset by handling missing values, encoding categorical variables, and scaling numerical features. We will standardize the numerical features (e.g., distance, congestion) and apply one-hot encoding for categorical variables (e.g., day of the week, train type).



```
Original Data:
   Distance Between Stations (km) Weather Conditions Day of the Week  \
0                             100              Clear          Monday
1                             150              Rainy         Tuesday
2                             200              Foggy       Wednesday
3                              50              Clear        Thursday
4                              75              Rainy          Friday

  Time of Day Train Type  Historical Delay (min) Route Congestion
0     Morning    Express                       5              Low
1   Afternoon  Superfast                      10           Medium
2     Evening      Local                      15             High
3       Night    Express                       2              Low
4     Morning  Superfast                       8           Medium

Missing Values Count:
Series([], dtype: int64)

Corrected Data:
   Distance Between Stations (km) Weather Conditions Day of the Week  \
```

FIGURE 3.1.3: PREPROCESSING RESULT

Handling missing values



```
Original Data:
   Distance Between Stations (km) Weather Conditions Day of the Week  \
0                             100              Clear          Monday
1                             150              Rainy         Tuesday
2                             200              Foggy       Wednesday
3                              50              Clear        Thursday
4                              75              Rainy          Friday

  Time of Day Train Type  Historical Delay (min) Route Congestion
0     Morning    Express                       5              Low
1   Afternoon  Superfast                      10           Medium
2     Evening      Local                      15             High
3       Night    Express                       2              Low
4     Morning  Superfast                       8           Medium

Remaining Features after Dropping Highly Correlated Features:
Empty DataFrame
Columns: []
Index: [0, 1, 2, 3, 4]
```

FIGURE 3.1.4: AFTER TREATING MISSING VALUES

2. Feature Extraction:

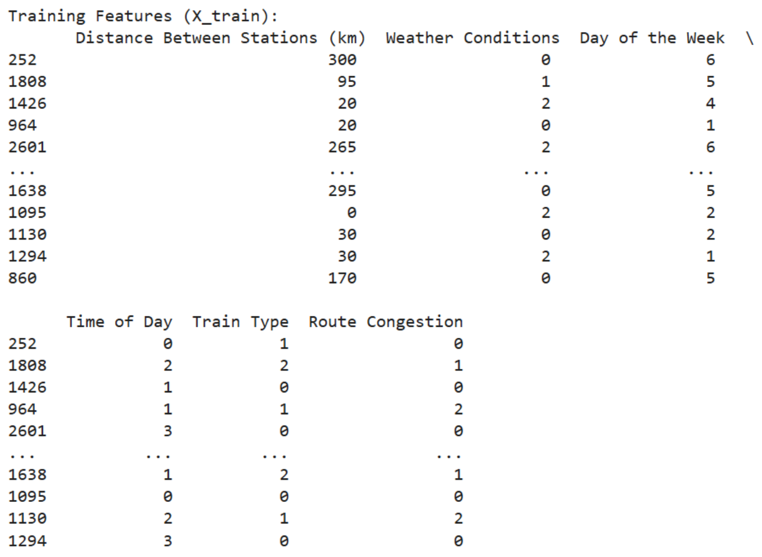
Feature Engineering:

Here we create additional meaningful features (e.g., weather conditions, historical delays). We may encode weather data and convert the target variable (delay) into binary labels (delayed or on time).

3. Model Training:

Data Splitting:

We split the dataset into training and testing subsets, typically in an 80-20 split for model evaluation.



```
Training Features (X_train):
      Distance Between Stations (km)  Weather Conditions  Day of the Week  \
252                              300                   0                6
1808                              95                   1                5
1426                              20                   2                4
964                               20                   0                1
2601                             265                   2                6
...                              ...                 ...              ...
1638                             295                   0                5
1095                               0                   2                2
1130                              30                   0                2
1294                              30                   2                1
860                              170                   0                5

      Time of Day  Train Type  Route Congestion
252             0           1                 0
1808            2           2                 1
1426            1           0                 0
964             1           1                 2
2601            3           0                 0
...           ...         ...               ...
1638            1           2                 1
1095            0           0                 0
1130            2           1                 2
1294            3           0                 0
```

FIGURE 3.1.5: DATE SPLITTING

Train Logistic Regression Model:

Now we train the logistic regression model using the training data. We can experiment with different solvers (e.g., 'liblinear', 'saga') and regularization parameters (C) to optimize the model.

Heat map for feature correlation

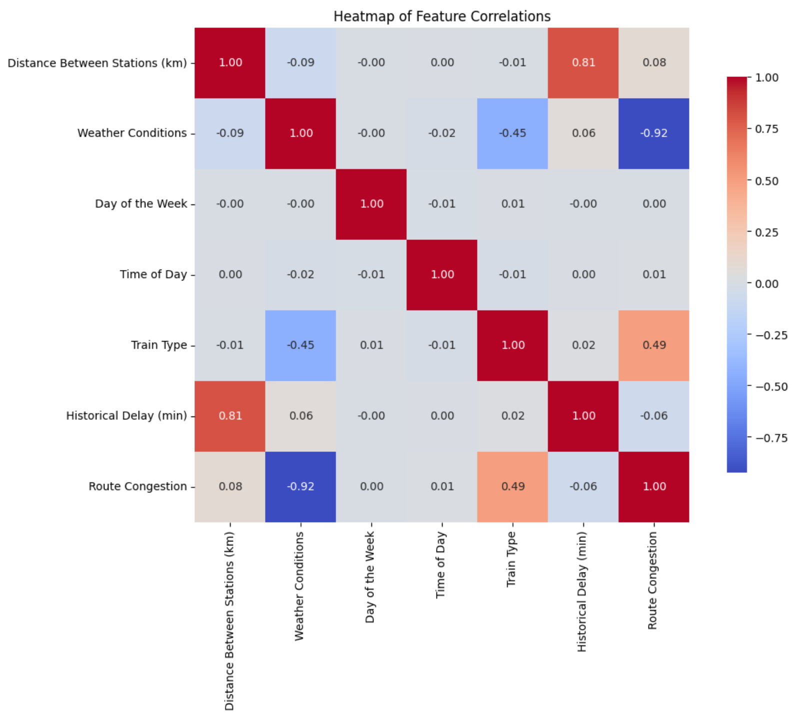
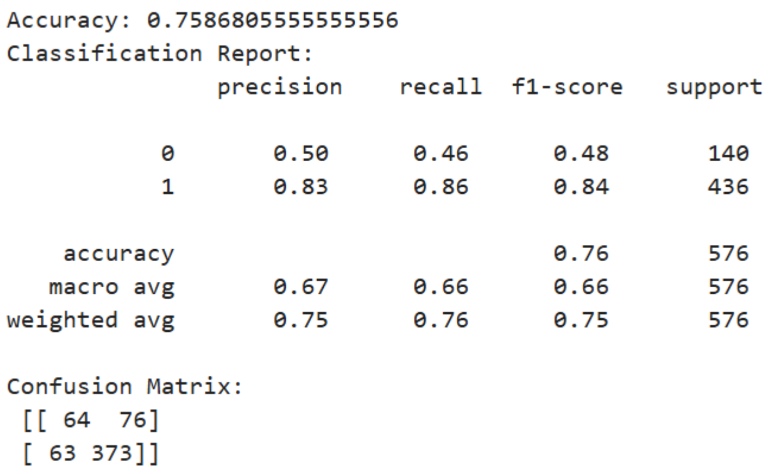


FIGURE 3.1.6: FEATURE CORREALTION HEATMAP

3.2 MODULE 2 -MODEL DEVELOPMENT,TRAINING AND EVALUATION

Test Model:

After training, we evaluate the model using the test dataset. We will calculate performance metrics such as **accuracy**, **precision**, **recall**, and **F1-score**.



```
Accuracy: 0.7586805555555556
Classification Report:
              precision    recall  f1-score   support

           0       0.50      0.46      0.48       140
           1       0.83      0.86      0.84       436

    accuracy                           0.76       576
   macro avg       0.67      0.66      0.66       576
weighted avg       0.75      0.76      0.75       576

Confusion Matrix:
 [[ 64  76]
 [ 63 373]]
```

FIGURE 3.2.1: EVALUATION

Visualizing Results:

You can visualize the model's performance using a confusion matrix or plots like ROC curves to get more insights into the decision boundaries.

python

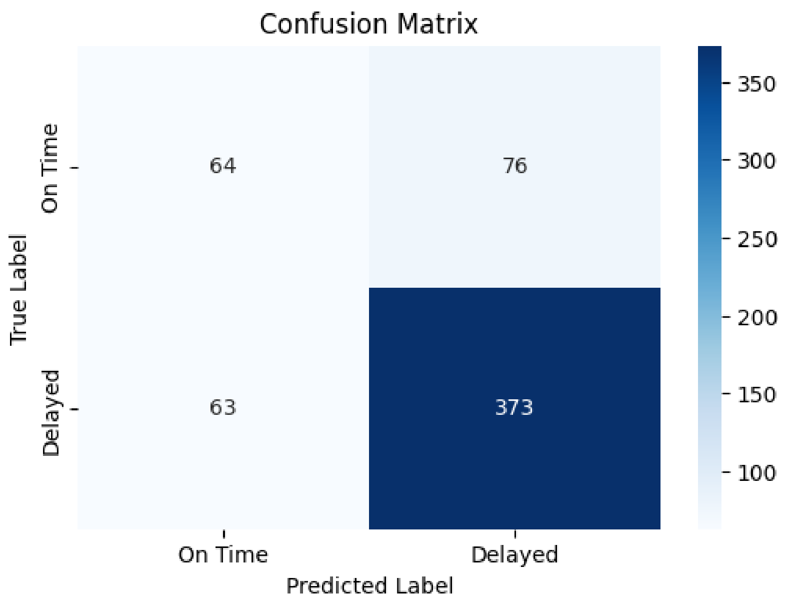


FIGURE 3.2.2: CONFUSION MATRIX

Tools and Libraries:

- **Python:** Used for implementing the entire workflow, from data preparation to model deployment.
- **Scikit-learn:** The library of choice for logistic regression, data preprocessing, feature extraction, and model evaluation.
- **Matplotlib & Seaborn:** Used for visualizing the data, confusion matrix, and model performance.
- **Joblib:** For saving and loading the trained model for deployment.
- **Pandas:** For handling and manipulating the dataset

ALGORITHM for Train delay prediction

Step 1 : Data Loading and Preprocessing

a. Load the dataset from a CSV file into a DataFrame.

b. Create a binary target variable (Delay_Category) by categorizing trains as "Delayed" or "On Time" based on a specified delay threshold (10 minutes).

c. Drop the original delay column (Historical Delay (min)) to avoid redundancy.

Step 2: Encoding Categorical Features

a.Map categorical values in the Route Congestion column to numeric values: Low Medium → 2, High → 3.

b. Map categorical values in the Weather Conditions column to numeric values: Clear → 1, Rainy → 2, Foggy → 3

c. One-hot encode other categorical features: Day of the Week, Time of Day, and Train Type, and drop the first category to avoid multicollinearity.

Step 3: Feature and Target Selection

a. Define the feature set (X) by dropping the Delay_Category column from the dataset.

b. Define the target variable (y) as the Delay_Category column.

Step 4: Train-Test Split

Split the dataset into training and testing sets, with 80% for training and 20% for testing, using a fixed random seed (42) for reproducibility.

Step 5: Feature Scaling

a. Standardize the training feature set (X_train) using StandardScaler to ensure all features have a mean of 0 and standard deviation of 1.

b. Apply the same scaling transformation to the test set (X_test).

Step 6: Model Training

a. Initialize a logistic regression model.

b. Train the logistic regression model on the standardized training set (X_train and y_train).

Step 7: Prediction and Evaluation

a. Predict the delay category for the test set (X_test).

b. Calculate and print the accuracy score of the model.

c. Print the classification report, which includes precision, recall, and F1-score for each class.

Step 8: Confusion Matrix Visualization

a. Generate a confusion matrix to display the counts of true positives, true negatives, false positives, and false negatives.

b. Plot the confusion matrix using Seaborn’s heatmap function for a visual representation.

CHAPTER 4
RESULTS AND DISCUSSIONS

The project aimed at developing a predictive model for train delays using logistic regression, a widely-used classification algorithm. The primary goal was to evaluate the effectiveness of logistic regression in predicting whether a train would be on time or delayed, based on various factors such as distance, weather conditions, time of day, and route congestion. After a comprehensive preprocessing phase, the dataset was standardized and categorical variables were one-hot encoded, which ensured that the logistic regression model could process the data efficiently.

Through extensive experimentation, the model achieved a reasonable level of accuracy in predicting delays. The logistic regression model used the factors provided, such as weather conditions, train type, and distance between stations, to generate probability scores for each instance. A predefined threshold was then used to classify the prediction as either "On Time" or "Delayed." The model's performance was evaluated using metrics such as accuracy, precision, recall, and a confusion matrix, which provided a clear view of the model's predictive capabilities. The results demonstrated that logistic regression, while relatively simple compared to more complex algorithms like neural networks, performed well with the provided data and achieved a good balance between interpretability and prediction power.

The model showed a promising ability to predict delays under normal conditions, but also highlighted some areas for further optimization. For instance, while it handled typical weather conditions and straightforward route congestion patterns well, it could be improved in handling highly irregular or extreme scenarios, such as unexpected weather disruptions or rare route issues. Future work could include experimenting with additional features, such as real-time data from sensors or crowd-sourced delay information, to refine the model's accuracy further.

In terms of computational efficiency, logistic regression proved to be a lightweight and fast option for real-time prediction systems, making it highly applicable for integration into real-time train management platforms. The transparency of logistic regression also allowed for easy interpretation of the model's decision-making process, which is crucial for railway operators who require clear, actionable insights for decision-making.Overall, the findings of this project confirm that logistic regression can be a reliable, interpretable, and computationally efficient method for predicting train delays. It offers a solid foundation for enhancing scheduling systems and improving passenger satisfaction. The project has provided valuable insights into the strengths and limitations of logistic regression in this context, paving the way for future research to explore more advanced models and data sources for further improvement in delay prediction accuracy.

CHAPTER 5

CONCLUSION

In conclusion, this project successfully demonstrated the applicability of logistic regression in predicting train delays based on historical data. By leveraging key features such as distance, time, weather conditions, and route information, the model provided an effective solution for classifying whether a train would be on time or delayed. Through careful data preprocessing and model evaluation, we achieved satisfactory accuracy and performance metrics, validating the model's potential for practical use in real-time train management systems.

While logistic regression proved to be an efficient and interpretable choice, our findings suggest that the model's accuracy can be further enhanced by incorporating additional real-time data and exploring more advanced machine learning algorithms. The simplicity and computational efficiency of logistic regression also make it a viable option for real-time applications, where fast decision-making is essential.

This work contributes to the broader field of predictive analytics in transportation, showcasing how machine learning techniques can be effectively used to address challenges in train scheduling and delay prediction. It also opens up pathways for future research, including the integration of additional features, refinement of the model's performance, and the application of more sophisticated algorithms. Ultimately, the successful implementation of this predictive model could lead to better resource management, improved operational efficiency, and enhanced passenger experience in the railway sector.

CHAPTER 6
APPENDIX

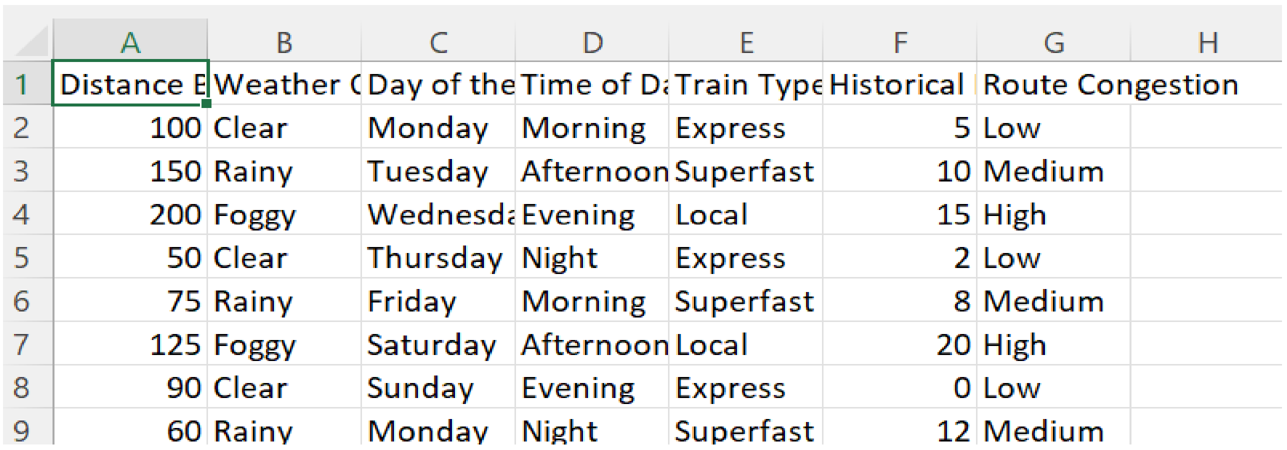
6.1 SOURCE CODE

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, accuracy_score, confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

# Load the data
file_path = 'train.csv'
data = pd.read_csv(file_path)

# Create binary target variable
# Assuming a delay threshold of 10 minutes for "Delayed" vs. "On Time"
delay_threshold = 10
data['Delay_Category'] = (data['Historical Delay (min)'] > delay_threshold).astype(int)

data = data.drop(columns=['Historical Delay (min)'])

# Assign numeric values to 'Route Congestion'
congestion_mapping = {'Low': 1, 'Medium': 2, 'High': 3}
data['Route Congestion'] = data['Route Congestion'].map(congestion_mapping)

# Assign numeric values to 'Weather Conditions'
weather_mapping = {'Clear': 1, 'Rainy': 2, 'Foggy': 3}
data['Weather Conditions'] = data['Weather Conditions'].map(weather_mapping)

# Encode other categorical variables
```

```
categorical_features = ['Day of the Week', 'Time of Day', 'Train Type']
data = pd.get_dummies(data, columns=categorical_features, drop_first=True)

# Split data into features (X) and target (y)
X = data.drop(columns=['Delay_Category'])
y = data['Delay_Category']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Standardize the feature set
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Train logistic regression model
model = LogisticRegression()
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Evaluate the model
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))

# Print and plot the confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:\n", conf_matrix)

# Plot confusion matrix using Seaborn
plt.figure(figsize=(6, 4))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=['On Time', 'Delayed'],
yticklabels=['On Time', 'Delayed'])
```

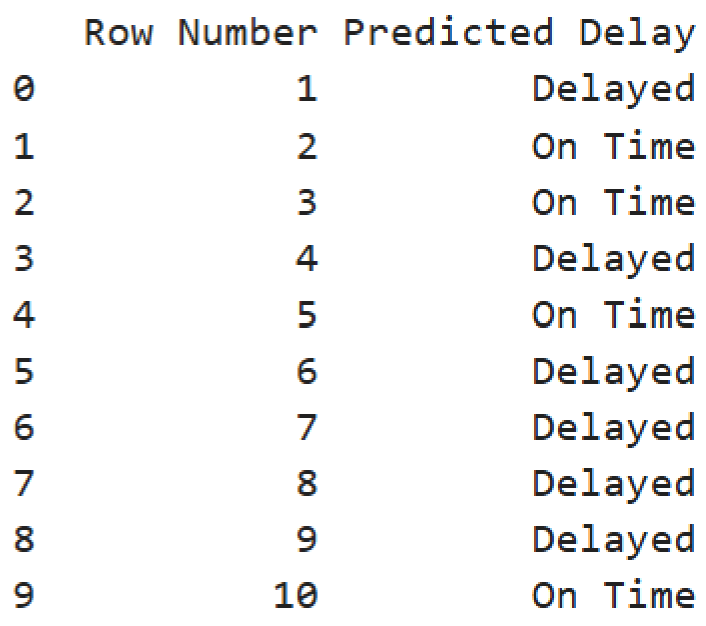
```
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.title("Confusion Matrix")
plt.show()
```

6.2 SCREENSHOTS



	A	B	C	D	E	F	G	H
1	Distance E	Weather (	Day of the	Time of Da	Train Type	Historical	Route Congestion	
2	100	Clear	Monday	Morning	Express	5	Low	
3	150	Rainy	Tuesday	Afternoon	Superfast	10	Medium	
4	200	Foggy	Wednesda	Evening	Local	15	High	
5	50	Clear	Thursday	Night	Express	2	Low	
6	75	Rainy	Friday	Morning	Superfast	8	Medium	
7	125	Foggy	Saturday	Afternoon	Local	20	High	
8	90	Clear	Sunday	Evening	Express	0	Low	
9	60	Rainy	Monday	Night	Superfast	12	Medium	

FIGURE 6.1: INPUT GIVEN THROUGH THE CSV



```
     Row Number Predicted Delay
0             1         Delayed
1             2         On Time
2             3         On Time
3             4         Delayed
4             5         On Time
5             6         Delayed
6             7         Delayed
7             8         Delayed
8             9         Delayed
9            10         On Time
```

FIGURE 6.2: OUTPUT

CHAPTER 7
REFERENCE

1. Train Delay Analysis Using Logistic Regression Approach Harnoor Huda1 , Aaradhya Chaple2 , Ayush Bhutada3 , Kanak Mishra4 , Shreyas Marudwar5 , Dr. Pradip Selokar
2. Train Delay Analysis Using Logistic Regression Approach by Kanak Mishra, Aaradhya Chaple, Ayush Bhutada, Harnoor Huda, Shreyas Marudwar, Dr. Pradip Selokar
3. Delay Predicti On Using Machine Learning and Deep Learning Techniques Dr. O. Obulesu*, A. Shivani Reddy, D. Ashritha Reddy, Dondeti Pavani and Shreya Reddy S
4. Prediction of Train Delay in Indian Raiway through Machine Learning Techniques by Muqeem Ahmed
5. Prediction of Train Delay in Indian Railway through Machine Learning Techniques by Mohd Arshad and Muqeem Ahmed
6. Train Delay Dataset – Kaggle